Administrivia/Last Time

- Reminders
  - Homework 3 due Thursday
  - Final on 5/6 at 4pm (room TBA)
    * Material included: Chapter 4 and onwards (rougly, second midterm material onwards)

- Topic list for “term paper” (for those not doing the final project) will be on class webpage this evening; paper due at project deadline

- Need a volunteer for SOOTs
Router Based Congestion Avoidance – DECBit

- Congestion Avoidance – Like TCP vegas, try to proactively reduce sending rate before hitting congestion
  - Congestion control is *reactive*; congestion avoidance is *proactive*

- TCP vegas did source-based congestion avoidance; can we have router based congestion avoidance?

- DEC bit: a router based congestion avoidance solution used in the Digital Network Architecture
  - Underlying network is connectionless, with a connection-oriented end-to-end protocol
  - Idea: router monitors average queue length, sets a bit on the packet if it is close to congestion
Random Early Detection

• Like DECbit:
  – Have RED routers monitor their queue length
  – Notify senders to slow down before congestion occurs

• Unlike DECbit – **notification is implicit**
  – TCP recognizes congestion when packets are dropped
  – Randomly drop packets before congestion occurs
  – TCP will learn about congestion through a timeout or a duplicate ACK

• What do you think; are we being too clever for our own good?

• What advantages/disadvantages over DECbit?
Router Role

• We have been assuming FIFO routers, is this the only way?

• What can the router do?
  – TCP with FIFO routers tries to maximize Power: ratio of throughput to delay
    • Try to send as much as possible without harming delay too much
  – Fairness: can the available resources be allocated fairly between the competing flows?
    • How about if we identify flows and use round robin?
    • Round robin not absolutely fair (variable packet sizes) – Fair Queueing (FQ) and Weighted Fair Queueing (WFQ) are used
Example – Real Time Applications

- “Timeliness” requirements must be met
  - Example: audio traffic

- Why not rely on the source to meet this requirement (as per the end to end argument)?

- Need more than best effort from the network
• Real time applications; if the data is late, it is worthless (not quite true)
• Buffering is used to hide the variability of the network
• Use a Playback buffer to nonuniformity of the network
• What QoS guarantees are needed?
  – Delay of no more than 300 msec (playback point)
  – Samples must be available at the receiver every 125 msec
  – Occasional Loss is Ok (fidelity requirement)
Each router has a queuing discipline to determine how packets are buffered and forwarded.

Queuing discipline affects performance and fairness:
- Bandwidth allocation (which packets to transmit)
- Buffer allocation (which packets to drop)
- Indirectly – latency (how long a packet waits)

We have been assuming: Single FIFO queue with drop tail – any problems?
Fair Queuing

- Idea: buffer traffic according to flows
- Allocate bandwidth according to flows
- What is a flow?
- Is this really fair?
- Is this even possible?
- What if we want differentiated services (more of the resources to go to one or more flows?) – weighted fair queuing (WFQ)
Fair Queuing

- Not all packets are the same; how to ensure fair share of the bandwidth?
- Can we do fine-grained sharing (e.g., bit level)?
- Idea: use a virtual clock to simulate fine sharing
  - Algorithm: Schedule packets depending on the time they are supposed to finish
- Preliminary/Definitions – For a single flow:
  - Let $P_i$ be the length of a packet $i$
  - Let $A_i$ be the arrival time of $i$
  - Let $S_i$ be the transmission start time of $i$
  - Let $F_i$ be the finish time for $i$
    * $F_i = S_i + P_i$
- When does a router start sending a packet
  - Assume clock ticks for every bit transmitted
  - If queue is empty, when the packet arrives ($A_i$)
  - Otherwise when $i - 1$ is done transmitting ($F_{i-1}$)
  - Therefore, the finish time is: $F_i = Max(A_i, F_{i-1}) + P_i$
Example – Multiple Flows

- Generalizing to multiple flows
  - Virtual clock is advanced 1 tick for every \( n \) bits transmitted if there are \( n \) active flows
  - Calculate \( F_i \) for each packet as it arrives on each flow
  - Send packet with minimum \( F_i \)
- Cannot preempt a packet that is already in transit
  - Think about the ATM cell size selection
- WFQ: give a flow a higher weight by making its clock tick slower
Example

Problem 6.13 in the book:

Consider a router that is managing three flows, on which packets of constant size arrive at the following wall clock times:

Flow A: 1, 2, 4, 6, 7, 9, 10
Flow B: 2, 6, 8, 11, 12, 15
Flow C: 1, 2, 3, 5, 6, 7, 8

All three flows share the same outbound link, on which the router can transmit one packet per time unit. Assume there is an infinite amount of buffer space. Arrival time ties are to be resolved in order A, B, C. At wall clock time $T=2$, FQ-clock time $A_i=1.5$. 
Quality of Service (QoS)

- Can the network provide guarantees on the “resource-related” service provided to applications?

- More than congestion handling, the network has to be aware of the application service requirements and honor them

- Some questions:
  - What happens if there are not enough resources?
  - What kinds of services are ...
    * Of interest to the applications and
    * Supportable by the network
  - How can this be actually implemented?
Applications

• Application Types
  – Rigid – fixed playback point – low jitter
  – Adaptive – application can vary playback point – jitter not an issue
  – Tolerant vs. intolerant to interrupted service/degradation

• Usually, its rigid and intolerant or adaptive and tolerant
QoS Contract

- For the network to provide a guarantee of service, it must know what the application requirements are.
- Fundamental change from best effort – need to be considered carefully.
- Major decision: the flexibility in describing the contract.
  - Fine-grained: per flow description (Integrated Services/IntServ)
  - Coarse-grained: classes of service (Differentiated Services/DiffServ)
- What is the tradeoff? Do coarse-grained schemes provide any guarantees for a given application?
  - How hard is it to implement either?
Implementing Integrated Services

- Need to be able to specify the requirements (contract)
- Need to be able to implement the contract
- Four Basic Aspects
  1. Resource Reservation: Request/implement the reservation
  2. Admission Control: can we support it?
     - In terms of bandwidth
     - In terms of delay
  3. Packet classification: separating the flows
  4. Packet Scheduling: implement the promises (+police abusers)
Mechanisms – Describing the Flow

- How should we describe a flow? Is the average rate sufficient?

- Some characteristics of a flow
  - Average transmission rate
  - “burstiness” indication
  - delay requirements
  - delay-variation tolerable
  - error tolerance
  - anything else?
Leaky Bucket Model

- Bucket leaks at a rate of $r$ packets per unit time
- Bucket can hold $B$ packets
- If packets arrive in a way that overfills the bucket, they are dropped
- Apply it to the flow examples above
- $r$ describes the average; $B$ the variability
Reference Implementation (RFC 1633)

- Classifier
- Packet Scheduler
- Admission Control
- Reservation Setup Protocol (RSVP)
Classifier

- IP must classify based on properties of the packet
  - Source/destination
  - Service indicators in IP header
  - Other? Flow id?

- Packets mapped into classes
  - A single flow (e.g., critical medical data)
  - A collection of flows
    * All video flows
    * All flows from organization X
Packet Scheduler

- Packets are queued for transmission at router or packet source
  - How is the transmission ordered?
    - Traditionally FIFO; ok for underloaded routers
    - Reorder packet queues to meet QoS guarantees (this is difficult; a lot of handwaving here)
    - Enforce (police) agreed upon flow characteristics
Packet Scheduler

- Guaranteed
- Guaranteed
- Predictive
- Predictive
- Best effort

- How to carry out the scheduling?
  - What are the constraints; is it the same for the three classes?
Important Theoretical Result [Parekh92]

- If you know the nature of the traffic source (e.g., leaky bucket)

- By implementing WFQ scheduling, you can get an absolute upper bound on the network delay of the traffic in question

- To simplify, we can derive upper bound on packet latency in the router
  - This allows us to implement QoS on delay
Levels of Service

• What types of guarantees should be provided by the network?
  – Guaranteed service: provide perfectly reliable upper bound on delay
    * Absolute guarantee on the upper bound of cumulative delay
  – Predictive service: provide “fairly reliable” (but not perfect) guarantee on upper bound
    * This kind of service is sufficient for our audio example
    * Think implementation – why might this be desirable?

• Best Effort
Scheduling

- Must provide delay guarantees for Guaranteed traffic
- Uses Weighted Fair Queueing to take advantage of the theoretical result above
  - Each guaranteed flow is treated as a separate input queue
  - Other services classes combined in one queue
  - Adjust weights (dynamically) to provide delay guarantees
- Predictive is implemented using simple priority
- Best effort traffic comes last
Admission Control

- Can a new flow be granted requested QoS without impacting earlier guarantees?

- Does resource-reservation request satisfy local administrative policies?

- Is requestor authorized to make reservation?

- Implementing reservation
  - Assuming the worst case use of all currently admitted flows
  - Proposal: assuming the average use of admitted flows
  - Proposal: measure the actual use of the flows and make decisions based on that
Resource Setup Protocol

- Resource ReSerVation Protocol (RSVP)
- Uses flowsspecs to describe flows
- Sender sends flow specification to receivers
- Receivers respond with reservation request
  - May reserve less than flow spec (e.g., low-quality audio is sufficient even though high-quality is transmitted)
  - May reserve more than flow spec (to receive from multiple senders)
  - Request travels back to the sender
  - Routers along the path reserve appropriate bandwidth and buffers
  - Routers merge reservation paths
Mechanisms – Implementing the reservation

- Two reservation messages: PATH and RESV
- A soft state is maintained by participating routers when the RSVP reply (RESV) packet is received
- What happens if the route changes?
- State is leased and must be renewed every 30 seconds
- Non participating routers are still best effort
- Support for Multicast QoS – will not go into it
Flowspecs

- Flowspecs: the user’s definition of the service required by the flow
- Consists of two decoupled specifications
  - Rspec: describes the service required from the network (e.g., predictive or guaranteed delay/delay target)
  - Tspec describes the flow’s traffic characteristics
    * Leaky bucket used for Tspec
    * What are the implications on the resources needed?
Differentiated Services

- Quality of service associated with aggregates instead of a single flow
- Routers can then deal with these aggregates differently according to their service requirements
- Example: IP’s expedited forwarding option ("premium service")
  - EF packets placed in a separate queue
  - Can use WFQ or simple priority to schedule them
- Police to some maximum peak rate, and drop offending packets
  - Is it practical to guess the “shape” of the aggregated traffic?
- Much more scalable than IntServ
- What happens to flows within an aggregate; will they be able to meet their service requirements?
Example DiffServ: Controlled Load

- Service guarantee: network looks “lightly loaded” for conforming traffic
- Assured Forwarding (RFC 2597)
  - Four Independent traffic classes
  - Three drop preference levels within each class
- Policing: Police to a specified rate and burst profile
  - Mark out of profile packets to have higher drop probability